Available online at www.sciencedirect.com
SCIENCE DIRECT
Computers and electronics in agriculture
ELSEVIER
Computers and Electronics in Agriculture 38 (2003) 217–225
www.elsevier.com/locate/compag

Application note

Agricultural chemicals use data access using COLDFUSION markup language and a relational database

Yulu Xia [a,*], R.E. Stinner [a], Daryl Brinkman [b], Norman Bennett [b]

[a] *North Carolina State University, 1017 Main Campus Drive, Suite 1100, NCSU Centennial Campus, Raleigh, NC 27606, USA*
[b] *National Agricultural Statistics Service, USDA, Washington, DC, USA*

Received 10 August 2002; received in revised form 15 November 2002; accepted 28 November 2002

Abstract

A relational database was developed for the agricultural chemical use data collected by the US Department of Agriculture, National Agricultural Statistics Service since 1990. COLDFUSION Markup Language was used for the client-side interface and server side process programming. The database is accessible from the Web at URL: http://www.pestmanagement.info/nass. Users can obtain information about agricultural chemical use in the database by search of crop, year, region, and active ingredient. Various agricultural chemical usage statistics are provided as Web tables, dynamically generated US maps, charts and graphs, and downloadable Excel files. We used a centralized software architecture in this project, which is suitable for projects with moderate programming complexity. A distributed approach might be more appropriate for the more complex projects. The current database information, spanning 1990–2001, will be augmented in the future, possibly using an automated updating scheme.

Keywords: Database; SQL; COLDFUSION; Dynamic access; Agriculture; Pesticide; Agricultural chemicals

* Corresponding author.
E-mail address: yulu_xia@ncsu.edu (Y. Xia).

0168-1699/03/$ - see front matter
doi:10.1016/S0168-1699(03)00003-6

1. Introduction

The United States Department of Agriculture (USDA) National Agricultural Statistics Service (NASS) conducts hundreds of surveys and prepares reports covering virtually every facet of US agriculture–production and supplies of food and fiber, prices paid and received by farmers, farm labor and wages, and other farm aspects of the industry (http://www.usda.gov/nass). NASS has also been surveying the use of pesticides (including biopesticides) and commercial fertilizers on farms in support of the President's Water Quality Initiative and the USDA's Food Safety Initiative since 1989. Statistics on field crops are published each year, while information on either fruits or vegetables is released in alternate years (http://www.usda.gov/nass/nassinfo/programs.htmchemical).

Before the chemical program was instituted, statistically reliable and readily available information on the amount and types of chemicals used in agriculture was quite limited. Consequently, neither USDA nor other concerned parties could respond adequately to questions of agricultural chemical use and its possible effects on food safety, water quality, and other environmental concerns. NASS has been publishing these data in various text formats. It is obvious that a searchable database would provide an invaluable tool to federal agencies in assessing the benefits and risks of pesticide use, to state agencies in determining how well actual agricultural practices accord with environmental quality standards, and to the public in understanding agricultural chemical usage in a state or region. Moreover, a database is essential to the sound evaluation of existing and proposed programs and policies that could affect food production, consumer prices, and farm income.

1.1. Relational database technology

Relational database technology is the dominant database technology currently used. The technology has been used extensively for developing information systems in agriculture (Jensen, 2001; Stafne et al., 2001; Huaman et al., 2000; Feidt, 1996). All major database management systems (DBMS), such as ORACLE, DB2, and SYBASE, are primarily relational database management systems. The main strengths of a relational database are simple data models and flexibility for future change.

A relational database is a collection of relations. Each relation is depicted as a table. Columns (usually called fields) in a table are attributes. Rows in a table represent entities (usually called records). Each record in a table can be uniquely identified by one or more fields. This field(s) is called the primary key. The field that is the primary key of another table is called the foreign key. All tables in a database are linked together by pairs of primary/foreign keys.

American National Standards Institute's STRUCTURED QUERY LANGUAGE (SQL) is the standard language for creation and manipulation of relational databases. SQL can be divided into a data definition language and a data manipulation language. A data definition language allows database developers to define a database and its tables. Then, the data manipulation language allows the end users to query and modify a database.

Another less common used database technology is called object database (also object-oriented database). Object databases are made of objects and classes. Objects are used to model real-world entities. A class is used to define a group of similar objects with the same data structure and the same operation. Object database technology is especially suitable for some biological systems (Xin et al., 2001; Beck et al., 1994; Beck, 2001; Congleton and Farrar, 1996). Details about object database can be found at Data Object Management Group (2002) and Chaudhri and Zicari (2001).

1.2. COLDFUSION markup language

The concept of a markup language is not new, even to the people who are not in the field of information technology. The majority of web pages these days are written using a markup language called HYPERTEXT MARKUP LANGUAGE (HTML). Another markup language called EXTENSIBLE MARKUP LANGUAGE (XML) is the standard language for describing and structuring data for exchange through the Web. All markup languages use tags for programming. These tags are either predefined (defined by the language or vendor like HTML) or defined by the developer such as XML.

COLDFUSION Markup Language (COLDFUSION hereafter) is developed by Macromedia. COLDFUSION is mainly used as a server side scripting language, which is executed by the COLDFUSION application server (Forta, 1998). COLDFUSION works by executing CFML templates, a mixture of HTML and COLDFUSION instructions that interface with databases and return the results as HTML to the requesting browser. Often this will include retrieving data from a database and generating web pages dynamically. COLDFUSION works in a manner similar to Java server pages or active server pages. The first COLDFUSION application server was developed in 1995. COLDFUSION is currently in version 6.0. COLDFUSION software supports many different web servers (such as Apache and IIS, on many different platforms such as SOLARIS, HP-UX, LINUX, and WINDOWS) and DBMS (such as ORACLE, SYBASE, SQL SERVER, ACCESS, FOXPRO, and SQL).

2. NASS agricultural chemical use relational database

The NASS Agrochemical Use Database consists of eight tables, based on data category. Following is a brief description of each of these tables.

RtUnit	contains information about commodity groups such as field crops, vegetables, fruits, etc, and the numerical code for each unit description
ProgramST	contains information about States in a chemical use program for a specific commodity (crop), chemical, and sample year

AIInfo	contains information associated with each active ingredient, including the full chemical name, short name, code, and class (fertilizer, herbicide, biological, insecticide, etc) of an active ingredient
CUData:	contains the actual survey quantitative data compiled at State level. These data include: (1) percentage of crop treated with a chemical, (2) the number of applications of active ingredient during a crop year, (3) amount of active ingredient applied per application, (4) amount of active ingredient applied per acre during the crop year, and (5) total amount of active ingredient applied during a crop year
CommInfo	contains information about each crop such as crop name, code, crop group etc
STName	contains State information–the name of each State, State code, and postal abbreviation
AIEPANum	contains information about active ingredient, including EPA registration number, chemical code, class, etc
Prod	contains information about agrochemical products such as product names and EPA registration number for each active ingredient

Data in the system were aggregated at the State level to ensure confidentiality of individual surveys. Because survey techniques vary among crops and years, significant effort was required to convert much of the raw data for internal consistency of all statistics. The database was designed and implemented by NASS. The National Science Foundation (NSF) Center for Integrated Pest Management maintains it using MICROSOFT SQL SERVER as the DBMS. Metadata of the database can be found at http://cipm.ncsu.edu/nass/metaData.txt.

3. The user interface and server side programming

A typical web application consists of three layers with the layers built on each other (Fig. 1; Wutka, 2000).

Data layer	Manages the data used by the application. This usually means data storage, DBMS, or EIS (Enterprise Information System)
Business logic layer	Contains various business rules and operations that the application performs on the data. In our case, these are the programs that interact with the database directly
Presentation layer	Interacts with the user in one way or another. This is typically a Web page or graphical user interface. People use the concept of tier to describe how a software application is structured in term of these layers
Single-tiered applications	Combine all three layers into a single component (usually an executable program)

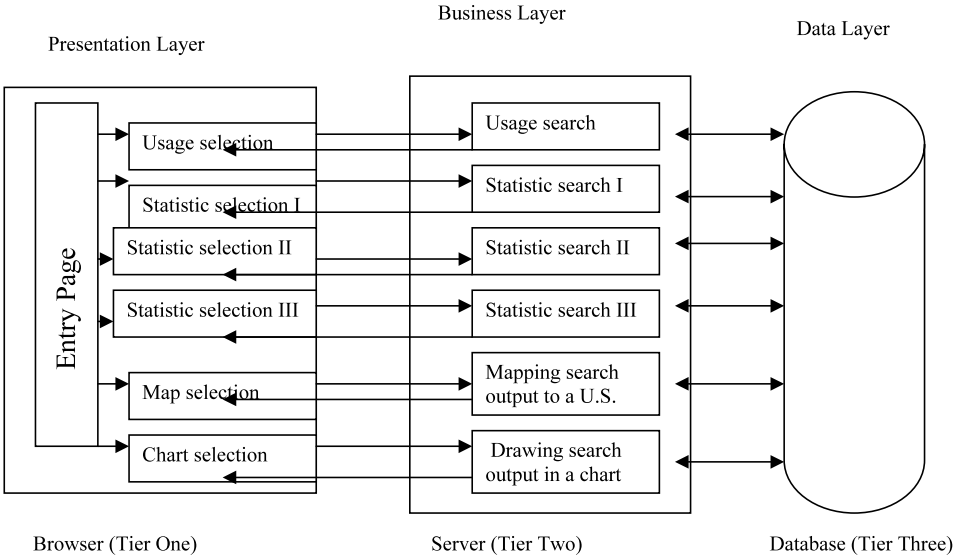

Fig. 1. Three-tiered architecture for this application.

Two-tiered applications	Usually combine the presentation and business logic layers into a single component and a separate database as the data layer
Three-tiered applications	Separate the three layers into separate components. If the application is distributed, it typically uses distributed object middleware such as Common Object Request Broker Architecture, Java Remote Method Invocation, Distributed Component Object Model, or Simple Object Access Protocol. A distributed application means that some components of an application are located in different computers across the network. For example, a web application may include a computing program that is located on another company's machine. You submit data from the application web site, it passes the data to where the computing program is located, and eventually returns the computation result back to the original site. Distributed applications help to reduce time and cost for software development or procurement
N-tiered applications	Similar to three-tiered applications, but they are more distributed than the three-tiered applications. Architecture such as Java 2 Enterprise Edition is an example of this type of application

We use a 3-tiered architecture in this project (Fig. 1). The presentation layer consists of seven subcomponents. The first component is the main page where users enter the system. From this page, a user can go to one of six search strategies

(subcomponents). These six search choices are (1) search the agricultural chemical usage data; (2) search usage statistics by year; (3) search usage statistics by state; (4) search usage statistics by commodity; (5) generate a US map based on usage statistics; (6) generate a chart based on historical usage data.

Once the user makes the selection and presses the search button, the business logic layer executes. The corresponding server side program queries the data layer (database) based on the user's selections, gets the query results, formats it in appropriate HTML, EXCEL, a map (Fig. 2), or a chart (Fig. 3) based on user's choice, and sends the information required to the presentation layer. The server side programs use Microsoft's Open Database Connectivity programming APPLICATION PROGRAM INTERFACE for accessing the database.

4. Discussion

Thanks to progress in information technology, it has become possible to develop a database-driven Web application like this within a reasonable time frame and limited financial resource. Unfortunately, this does not mean that development of a Web application is routine. Roughly 70% of all software projects fail (Helm and Quarto-vonTivadar, 2002). To ensure success, many basic technical and economic issues should be considered before embarking on a software project (Xia et al., 2002b). Once the basic issues are resolved, one may consider the issues such as software architecture as we mentioned earlier. For complex software applications, a distributed approach is usually the choice due to the benefits already described.

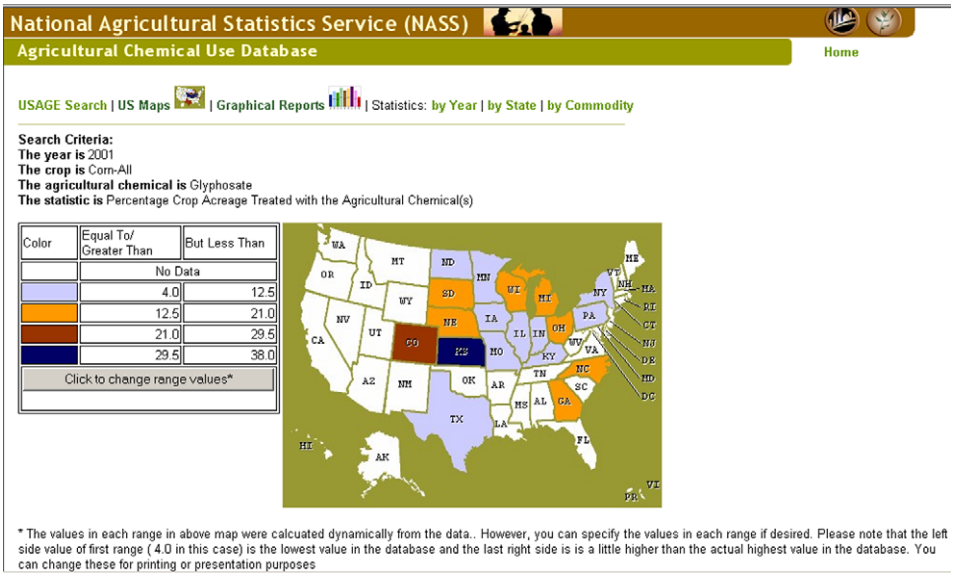

Fig. 2. Search output presented in a US map.

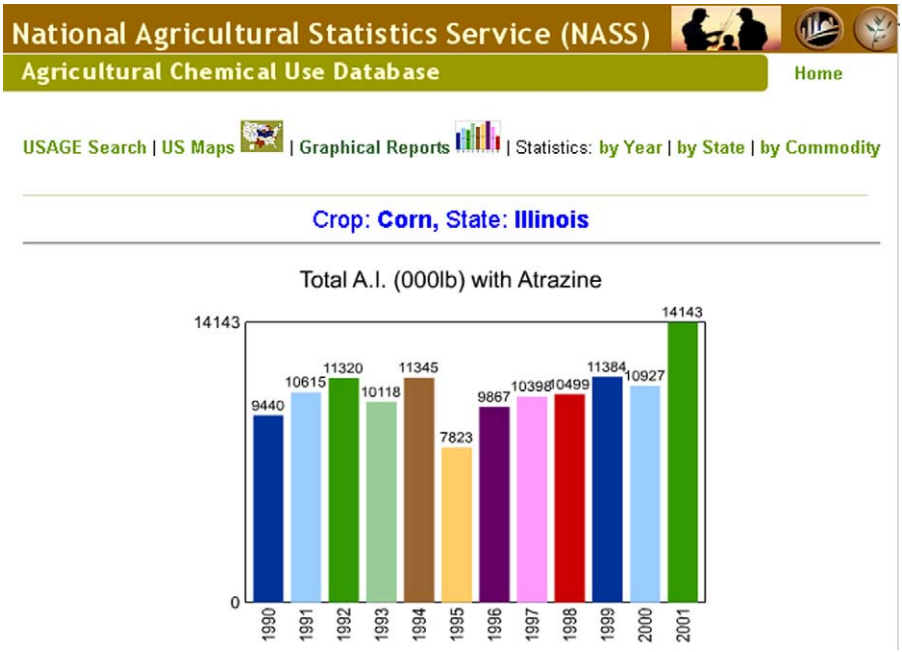

Fig. 3. Search output presented in a bar chart.

However, this kind of application relies more on programmer's skills, availability of software components, compatibility of related software tools, and the organization's Internet firewall settings. Any of above factors can slow down, or in the worse case, cause a project to fail. Hopefully, these issues will become less problematic with the recent adoption of Web service architecture (Xia et al., 2002b).

We used a centralized architecture for this application, wherein all software components are located either in a single machine or in different machines within an intranet. A centralized approach is the mainstream architecture for small- to medium-sized Web applications such as the one presented here. The main advantage of a centralized architecture is that the owner controls all components of a project. Consequently, it is easier to maintain and manage the project compared with a distributed approach. A centralized approach usually requires less programming skill than for a distributed architecture. However, it may take longer to complete a centralized project and require more financial investment with increasing software size and complexity. Another weakness of a centralized approach is that the power of a centralized software application can be limited due to the fact that all components in the project have to be developed in-house. As an alternative, developers can start a project as a centralized application, and gradually change to a distributed application as more functions are added to the project later.

Besides COLDFUSION, other general propose programming and scripting languages can also be used for web-based database applications. Mewes et al. (1997), Xin et al. (2001) and Beck (2001) used JAVA programming language in their projects.

JAVA is generally considered a more complex and powerful language than COLDFUSION. Other commonly used languages (technologies) include ACTIVE SERVER PAGE, which usually uses VISUAL BASIC as the programming language and a hypertext preprocessor. For more information about the technologies, please refer to Xia et al. (2002a).

Our database contains information from 1990 to 2001. Additional data will be added as they become available. Currently, NASS sends new data to the NSF Center for Integrated Pest Management by e-mail and we parse the data into the database. We are exploring development of software to automate the transfer of data using XML and JAVA. This new software will also help us provide new transport mechanisms for data sharing with other interested parties.

The survey data are all stored at the state level. There is only one data point for a given crop and agrochemical during a given year. Therefore, it is impossible to search for ranges and means by all three criteria in a single search. We designed the 'statistical search' to allow users to select either (Year and Agrochemical), (State and Agrochemical), or (Crop and Agrochemical). The search returns three statistical values (maximum, average, and minimum) for each of the agrochemical use statistics, such as percentage treated and active ingredient applied during a crop year.

This information system is now operational and publicly available. This project provides a value tool to aid in decision making about agrochemical uses at the federal and state level. It also gives researchers, educators, and the public a reliable and convenient access to pesticide use data developed by NASS.

Acknowledgements

This project is funded by USDA, The Cooperative State Research, Education, and Extension Service (CSREES), project award No. 2001-34366-10324.

References

Beck, H., 2001. Agricultural enterprise information management using object databases, JAVA and CORBA. Comput. Electron. Agric. 32 (2), 119–147.

Beck, H.W., Gilman, E.F., Fowler, P.A., 1994. An expert system for tree selection in urban forestry. Appl. Eng. Agric. 10 (5), 743–747.

Chaudhri, A.B., Zicari, R., 2001. Succeeding with Object Databases: a Practical Look at Today's Implementations with JAVA And XML. Wiley, New York, NY.

Congleton, W.R., Farrar, L.M., Jr, 1996. An object-oriented database with ancestry for controlling inbreeding. AI Appl. 10 (2), 51–61.

Data Object Management Group, 2002. Standard Overview. http://www.odmg.org/standard/standardoverview.htm.

Feidt, W.B., 1996. Building a W3 scientific database: database of the occurrence and distribution of pesticides in Chesapeake Bay. Agric. Lib. Inf. Notes 22 (9/12), 1–7.

Forta, B., 1998. The COLDFUSION Web Application Construction Kit, third ed. Que, Indianapolis, IN, p. 1998.

Helm, H., Quarto-vonTivadar, J., 2002. Discovering Fusebox 3 with COLDFUSION. Techspedition, Las Vegas, NV.

Huaman, Z., Hoekstra, R., Bamberg, J.B., 2000. The inter-genebank potato database and the dimensions of available wild potato germplasm. Am. J. Potato Res. 77 (6), 353–362.

Jensen, A.L., 2001. Building a web-based information system for variety selection in field crops—objectives and results. Comput. Electron. Agric. 32 (2), 195–211.

Mewes, H.W., Albermann, K., Heumann, K., Liebl, S., Pfeiffer, F., 1997. A database for protein sequences, homology data and yeast genome information. Nucleic. Acids Res. 25 (1), 28–30.

Stafne, E.T., Brown, J.S., Shine, J.M., Jr, 2001. A relational database for agronomic and genealogical sugarcane data: an adaptable prototype. Agron. J. 93 (4), 923–928.

Wutka, M., 2000. Using JAVA Server Pages and Servlets. Que, Indianapolis, IN.

Xia, Y., Wang, X., Stinner, R.E., 2002a. Simple Object Access Protocol (SOAP) for Bioinformatics Software Interoperability.Bioinformatics 1, 74–82. URL:http://www.cpb.uokhsc.edu/ojvr/soa-pabs2002.htm.

Xia, Y., Stinner R.E., Chu, P., 2002b. Database integration with the Web for biologists to share data and information. Online J. Biotech. http://www.ejb.org/content/vol5/issue2/.

Xin, J., Beck, H.W., Halsey, L.A., Fletcher, J.H., Zazueta, F.S., Momol, T., 2001. Development of a distance diagnostic and identification system for plant, insect, and disease problems. Appl. Eng. Agric. 17 (4), 561–565.